Séance 4: PROCESSING

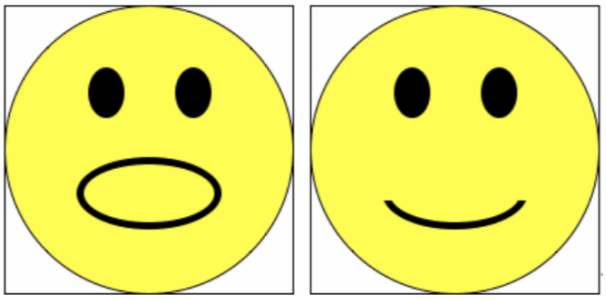
L1 – Université Nice Sophia Antipolis

Objectifs:

- Utilisation de Processing (avec Python)
- Apprendre à dessiner des formes géométriques
- Gérer une animation avec du texte
- Dessiner avec Processing

Processing Dans ce TP nous allons utiliser Processing pour faire des dessins, des animations, et même des jeux ! Oubliez IDLE, dans ce TP nous allons travailler directement avec l'environnement de travail de Processing. Lancez Processing puis vérifiez que le mode Python a déjà été installé : un onglet en haut à droite affiche Python si tout va bien, sinon il affiche probablement Java. Cliquez sur cet onglet pour installer le mode Python au besoin. Revenez à la fenêtre d'édition du « sketch ». C'est parti ! [1]

Exercice 1 (Souriez, ⋆)



Cet exercice a pour but de construire un « smiley ». Vous allez le faire en plusieurs étapes.

1. *Définissez les couleurs blanc, jaune, et noir que vous allez utiliser. Rappelez-vous, une couleur est obtenue par mélange de rouge, vert, et bleu, chacun dans une quantité variant entre 0 et 255.* [2]

```
noir  = color(0, 0, 0)
blanc = ...
jaune = ...
```

2. *À l'aide de la fonction* `size`*, fixez les dimensions 300×300 de votre image. Ajoutez un fond blanc à l'aide de la fonction* `background`*.*
3. *Choisissez la couleur de contour noir (fonction* `stroke`*) et la couleur de remplissage jaune (fonction* `fill`*). Choisissez le mode de coordonnées centré (*`ellipseMode(CENTER)`*). Tracez un disque jaune de rayon 270 placé au centre de la fenêtre (fonction* `ellipse`*).*

2. Vous pourrez vous aider d'une pipette à couleurs (color picker), par exemple https://rapidtables.com/web/color/RGB_Color.html.

4. *Ajoutez les yeux, qui sont des ellipses remplies de noir centrées au 2/3 de la tête dans le sens de la hauteur, et à 1/3 et 2/3 dans le sens de la largeur ; la largeur de chaque ellipse sera 20 et sa hauteur 30.*
5. *Ajoutez la bouche ouverte à votre image en dessinant une ellipse d'épaisseur de trait 5 (fonction* `strokeWeight`*) Puis masquez la partie haute de la bouche avec un rectangle jaune (fonction* `rect`*)*
6. *Bravo ! Vous pouvez signer votre oeuvre (fonction* `text`*).*

□

Exercice 2 (Filmez, $\star$)
En reprenant le code de l'exercice précédant, vous allez créer une animation où le smiley cligne de l'oeil gauche à intervalle régulier.

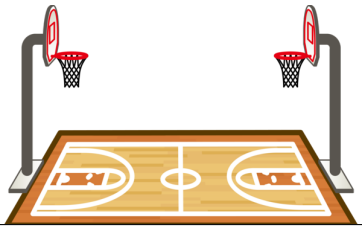
1. *Commencez par structurer votre code : vous allez créer trois fonctions dont deux sont des* fonctions réservées *(cf code ci-dessous). Il vous faut donc répartir le code que vous avez écrit dans les bonnes parties suggérées ci-dessous. Indication : la majorité du code va se retrouver dans la fonction* `dessine`*, et pour le moment vous n'avez rien à rajouter à* `draw`*.*

```
# variables globales
noir = color(0, 0, 0)
...

def setup() :
    # fonction reservee
    # executeee une seule fois au debut
    size(300, 300)
    ...

def draw() :
   # fonction reservee
   # rappellee automatique a chaque top d'horloge
   dessine(30)

def dessine(h) :
   # dessine le smiley avec une hauteur h pour l'oeil gauche
   ...
```

2. *Retrouvez l'instruction qui dessine l'oeil gauche et faites-la dépendre du paramètre h de la fonction dessine. Vérifiez que votre dessin s'affiche toujours comme il faut.*
3. *Vous allez maintenant créez une animation en vous basant sur le modèle vue calcul (cf cours). Le monde sera ici une variable* `t` *qu'on incrémente à chaque top d'horloge.* `t` *sera initialisée dans* `setup` *et incrémentée dans* `draw` *; **n'oubliez pas d'ajouter*** `global t` ***au début de ces deux fonctions**. Il vous faut ensuite installer une horloge à 28 tops par seconde (fonction* `frameRate`*). Enfin, il va falloir définir la hauteur h de l'oeil gauche en fonction de t, de sorte qu'on oscille entre 0 et 30 périodiquement. Par exemple, prenez $h = 30sin(t\frac{\pi}{56})$ pour faire fermer et rouvrir l'oeil en 2 secondes. Si vous avez tout mis au bon endroit, votre ami devrait être en train de vous faire de l'oeil...*

□

Exercice 3 (La balle au bond, ⋆⋆)



Vous allez créer une animation avec un ballon de basket qui rebondit sur les bords de la fenêtre de dessin. Votre monde comportera la position du ballon de basket et son vecteur vitesse (voir code à trou ci-dessous). Vous allez construire votre animation pas à pas.

```
def setup() :
    global xpos, ypos, xvit, yvit # le "monde"
    global ballon, rayon, L, H
    size(640, 480)
    frameRate(28)
    imageMode(CENTER)
    # on initialise le monde
    xpos = 80
    ypos = 80
    xvit = 2.8
    yvit = 2.2
    # l'image du ballon
    ballon = loadImage("basket-ball.png") # l'image du ballon
    # cf http://deptinfo.unice.fr/ elozes/AlgoPython/basket-ball.png
    # puis a installer dans le repertoire data du sketch
    rayon  = 30 # le rayon du ballon (redimensionner l'image)
    # les dimensions de la fenetre
    (L, H) = 640, 480

def draw() :
    global xpos, ypos, xvit, yvit
    (xpos, ypos, xvit, yvit) = suivant(xpos, ypos, xvit, yvit)
    dessine(xpos, ypos)

def suivant(xpos, ypos, xvit, yvit) :
    # a modifier!
    return (xpos, ypos, xvit, yvit)

def dessine(xpos, ypos) :
    # a completer

```

1. *Recopier le code et compléter la fonction* `dessine`. *Vous devriez avoir une balle immobile sur fond blanc.*
2. *Complétez le code de la fonction* `suivant` *: à chaque top d'horloge, la position est mise à jour en translatant le point (xpos,ypos) selon le vecteur vitesse. Lorsque la balle touche le haut ou le bas de la fenêtre,* `yvit` *change de signe. De même, lorsque la balle touche les bords gauche et droits, c'est* `xvit` *qui change de signe. Attention, votre balle ne doit pas sortir de la fenêtre !*
3. *Vous allez maintenant rendre votre animation interactive : vous voulez pouvoir dévier la balle en utilisant les flèches du clavier ; par exemple, une pression sur la flèche vers le haut diminuera* `yvit` *de 0.1. Ajoutez la fonction réservée* `keyPressed` *à votre code (voir cours). Attention les yeux, ne dribblez pas trop vite !*
4. *Ajoutez la capture de la balle avec la souris (fonction* `mousePressed`*, cf cours) : lorsque vous cliquez sur la balle, elle se fige (le vecteur vitesse devient nul). Si vous cliquez à côté, rien ne se passe. Caught it ?*

□

Exercice 4 (Suivez mon regard, ⋆⋆−⋆⋆⋆)

Programmez le célèbre Xeyes (http://arc.id.au/XEyes.html), vous trouverez la photo de Dali sur http://deptinfo.unice.fr/~elozes/AlgoPython/Dali.gif.

L'idée de base de XEyes est la suivante : chaque fois que le curseur se déplace, les pupilles des yeux se déplacent dans la direction du curseur, dans les limites de l'orbite de chaque œil.

Il y a trois couches d'images :

- *en fond, une image (blanche) pour la couleur du blanc des yeux ;*
- *au-dessus, des dessins pour les pupilles des yeux (des cercles ou ellipses) ;*
- *au premier plan, une image pour le visage, avec les yeux transparents pour laisser visibles les blancs et pupilles placés dessous.*

On fixe une position « initiale » du centre de la pupille $I(x_0, y_0)$; le centre de la pupille pourra être placé dans l'orbite, c'est-à-dire dans le disque de centre $I(x_0; y_0)$ et de rayon r choisi à l'avance.

Pour placer le centre $P(x, y)$ de la pupille,

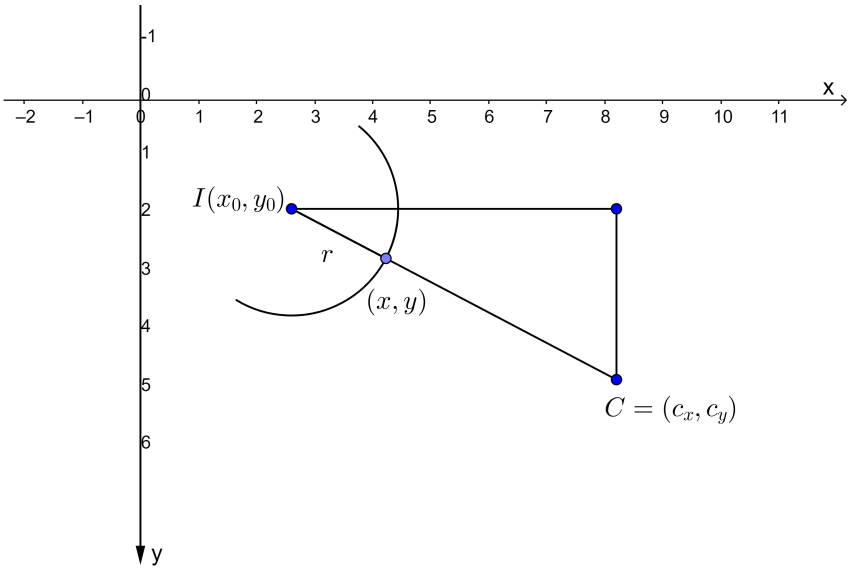
- *si le curseur est à l'intérieur de l'orbite, alors le centre de la pupille est la position du curseur ;*
- *sinon, le centre de la pupille est en direction du curseur, dans la limite de l'orbite.*

Le curseur a pour coordonnées $C(c_x, c_y)$. Les coordonnées $P(x, y)$ du centre de la pupille sont alors

$$x = x_0 + \frac{r}{IC}(c_x - x_0) \quad \text{et} \quad y = y_0 + \frac{r}{IC}(c_y - y_0)$$

La longueur IC est calculée grâce au théorème de Pythagore :

$$IC = \sqrt{(c_x - x_0)^2 + (c_y - y_0)^2}.$$



Indications :

- *On chargera l'image une seule fois. On pourra utiliser le code suivant.*

```
def setup() :
    global dali
    size(514, 800)
    dali = loadImage('Dali.gif') # on charge l'image une seule fois

```

Il suffira ensuite d'appeler `image(dali,0,0)` *pour l'afficher.*

— *On pourra définir une fonction* `oeil(x0,x1,cx,cy,r)` *pour éviter de répéter du code pour chaque œil.*

— *La valeur* r *est arbitraire, à choisir en expérimentant.*

— *Les coordonnées du curseur sont données par* `mouseX` *et* `mouseY`.

□